

DRAFT

Configuration Document for the Optronics 746 Spectroradiometer, used as the Calibration Facility Calibration Transfer Instrument

Author: John Marketon

Date: 21 June 1999

This document describes the state of the Optronics 746 Spectroradiometer. This instrument is located at the NASA GSFC Code 920.1 Calibration Facility, and is used as the CF calibration transfer instrument.

Keywords: 746, Calibration Transfer Instrument, Spectroradiometer

1.0 General Description

The 746 is a stand-alone system with three major components:

- A single pass monochromator in the Czerny-Turner configuration.
- A set of control electronics.
- A data acquisition computer.

There have been no changes to this system since delivery in early 1990. Thus, manufacturer's documentation remains valid, and sufficiently describes the current state of the system.

Manufacturer's documentation includes:

- Model 746(746-D) Infrared Spectroradiometer Operation Manual
- Model 740-1C/D Deluxe Wavelength Drive Operation Manual
- Model 740-14B Te Cooled PbS Detector Package Operation Manual
- Model 740-13 4-Inch Diameter Integrating Sphere Operation Manual
- Scientific Solutions Base Board Installation Manual & Users Guide

These documents are available in the CF Library.

2.0 Electronic Components

The primary components of the 746 calibration transfer instrument are:

- One monochromator containing filter, chopper, and grating mechanisms and drive electronics, in addition to the collimating and bending mirrors and a three-grating turret (Model 735IR)
- One lock-in amplifier with auto-ranging pre-amp (Model 736)
- One wavelength drive (Model 740-1C/D)
- One Pre-amp front end amplifier (Model 736)
- One ThermoElectric controller (Model 740-14B)
- One PbS detector head
- One Si detector head
- One IBM-PC compatible data acquisition computer containing one Scientific Solutions Base Board to interface with the lock-in amplifier and wavelength drive units.

DRAFT

3.0 Data storage format

The data storage format for the 746 system is described in detail in the CF document titled *746 Data Format*.

4.0 Software

A listing of the software used to control the 746 spectroradiometer is in Appendix A.

DRAFT

Appendix A – 746 Control Software Listing

```
-----
' Program Name: 746SIG-C
' Version Date: 05/17/99 (turbo basic v1.1)
' Written : 11/20/93
' 5/17/99 by John Cooper - Modified 746SIG-B and renamed to
' 746SIG-C. Put "StartOfData:" line in .ALL file header.
' Changed HardCopy default to "N".
' 11/20/93 by John Cooper - Modified to take a user requested number of
' samples per wavelength and save data in two files:
' *.ALL (all output from 746) and *.AVG (averaged data).
' Only signal data is saved;
' no system reponse or cal. input files are used.
' Averaged ptime mark in .ALL file at beginning of each
' wavelength scan.
' 12/23/93 JC - finished taking out all line numbers and adding
' labels.
' 12/27/93 JC - Rewrote signal collection to have min and max
' number of samples. New method takes min samples
' before checking noise, then takes up to max samples
' until noise is less than specified level.
' Renamed from 746SIG-A to 746SIG-B (2nd gen.).
' Corrected for rounding errors in noise check routine.
' (in BeginSampling)
' 1/3/94 JC - Made subroutines for "sound" segment and
' for hitting spacebar to continue.
'
' Old Changes (746ric)
' 11/12/87 by Bill Gorlitz , Bill Santspre -- added an integrating
' time routine and defined variable MSM# to add micro-
' processor independant time delays to measurement
' routines.
'
' 2/10/88 BY BILL SANTSPREE -- MODIFIED CODE TO ADAPT TO LATEST
' revision 740-1C/D to correct for intermittent filter
' wheel changes. Changed neutral command from 15 to 0.
' Changed error checking of data file to limitations of
' range as set by the filter grating file (GR-FL).
'
' 3/18/88 by Bill(s) -- Filter delay loops to fast for new filter
' Changed appropriate lines near beginning wavelength.
'
' 8-22-88 Corrected loop-back path for DATA NOT READY in
' Signal Read Sub so that delay is not included.
'
' 9/16/88 Replaced Timeloop Measurement loop with revised edition
' that is faster on all machines. Placed it in a subroutine.
'
' 9/16/88 Added a double check at the end of the Beginning Wavelength
' when Mono is stopped that tests for +/- .1 wavelength and if
' not then sends back to Beginning Wavelength. For false reads.
'
' 9/28/88 Added Ftime1 and Ftime2 time variables to replace filter
' shutter delay loop. 6 sec delay includes spacebar press.
'
' 3/22/90 Replaced errant GOTO's in subs with RETURN's.
' 4/03/90 Added units for radiance using cm's.
' 4/03/90 Compressed and centered units in HEADER.
'
' System Configuration: 746(746D), 740-1C/D, 746-410, IBM PC w/ Tecmar Base Board
'-----
' SCREEN 0, 0, 0, 0: WIDTH 80: CLS
' KEY OFF
'
' - INITIALIZATION -
'
' arrays
' WvLength#() : wavelengths
' AvgSgnl#() : system averaged output values
' NrmVal#() : normalized values
```

DRAFT

```
' FltrWvlngth#() : filter cut-on wavelengths
' GrtngPrm#() : grating parameters - blaze, grooves/mm, cut-on wavelength
' Sgnl#() : all samples per each wavelength
'
' strings
' AVGFILE$ : .AVG signals data file name
' ALLFILE$ : .ALL signals data file name
' HdrData$ : for entering extra header data
' DateEntry$ : date (yyymmdd format)
' TimeMrk$ : time mark at beginning of each wavelength scan
' K1$-K10$ : soft key labels
' Q$ : program directives
' Source$ : source designation
' Unit$ : system output data file units
' Dashes$ : for PRINT/LPRINT of 80 dashes
' Hardcopy$ : "Y" to print, "N" to not print anything
' WaitMessage$ : msg. displayed before requesting spacebar from user
'
' simple variables
'
' -variables for average signal output data file (.AVG) and .ALL file
' YMD& : data file date
' LowWav# : lower wavelength limit
' HiWav# : upper wavelength limit
' Intrvl# : data wavelength interval
' NumWvlngths% : number of spectral values
'   RmTemp : room temperature (c)
'   RelHumid : relative humidity (%)
'   SrcDst : distance to source (cm)
'   SlitSz : slit size (mm)
' NumSmpls% : minimum number of samples per wavelength
'             before noise check.
' MaxSmpls% : maximum number of samples per wavelength
'             if noise level is not reached
'
' I,J%,J0%,Y,X : counters, FOR-NEXT loop
' H% : counter, printout format (five times)
' V% : counter, run #
' FLG1% : flag FLG1%=1, date entered & set up displayed
' FLG2% : flag FLG2%=1, same grating type, det. type, noise level & designation
' FLG3% : flag FLG3%=1, scan aborted
' FLG5% : flag FLG5%=1, enter different data file
' Sampl% : sample counter for each signal scan ***
' Noise# : signal sampling %
' WaveRdng# : 740-1C reading
' WavTest# : temporary storage of test wavelength or MaxSgnl#'s wavelength
' WaveSlow# : (WavTest#-.3), wavelength at which motor switches to slow mode
' WaveChk# : (WavTest#-.4), rechecks wavelength before motor switches to slow mode
' WavFactr# : wavelength correction factor
' SumSgnls# : summation of data signals ***
' SumSqrSgnls# : summation of squared data signals ***
' SgnlOut# : 736 reading ***
' SgnlAvg# : signal average ***
' SgnlPctDev# : signal standard deviation / average ***
' FKey% : numeric expression for computed GOTO
' Tmpl% : temporary storage of a numeric value
' MaxSgnl# : stores larger of two compared data points
' YAxisInc% : Y-axis increment
' LocDot% : location of "." in AVGFILE$
' GrtLow#,GrtHigh# : lower,upper limits of mechanical range imposed by grating type
' WavForGrt# : wavelength correction factor imposed by grating type
' FltStart% : starting filter position
' FltCrrnt% : current filter positon
' GrtStType# : starting grating type
' GrtCurrent# : current grating type
' MSM# : FOR/NEXT loop milisecond multiplier based on system clock
' NumFltrs% : Number filter positions in filter wheel
' NumGrtngs% : Number of grating types being used in turret (Cannot be > 3)

GOSUB TimeLoopMeasurement
'
P1CW = 528: P1C = 529: P1B = 530: P1A = 531' assign numeric values for Tecmar
P2CW = 532: P2C = 533: P2B = 534: P2A = 535' port sections 1&2;base address=528
```

DRAFT

```
OUT P1CW, 147' Port A:in ; Port B:in ; Port C lower:in ; Port C upper:out
OUT P2CW, 147' Port A:in ; Port B:in ; Port C lower:in ; Port C upper:out
HIMASK = 240: LOMASK = 15' byte masks
'
OPTION BASE 1
DIM AvgSgnl$(501), WvLength$(501), NrmVal$(501), FltrWvlnth$(10), GrtngPrm$(3,
3), Sgnl$(100)
'
Dashes$ = "-----"
-----"
Hardcopy$ = "N"
WaitMessage$ = "Press spacebar to continue"
YAxisInc% = 1
RmTemp = 24: RelHumid = 30: SrcDst = 50: SlitSz = 2.5
NumSmpls% = 5 : MaxSmpls% = 20
V% = 0: FLG1% = 0: FLG2% = 0: FLG3% = 0: FLG5% = 0
'
' - START -
'
PRINT "746(746-D) System Signal Output Calibration Program"
PRINT: PRINT
'
' - INITIALIZE/SET-UP -
' - MEASUREMENT PARAMETERS -
'
' - GRATING/FILTER TYPE -
'
IF FLG5% = 0 THEN LoadFltGrtFile

GrFileLoaded:
CLS
FLG5% = 1
IF Tmpl% = 5 THEN DateEntry
CLS
PRINT "CURRENT CONFIGURATION"
PRINT "Data File Name: "; ALLFILE$
PRINT "Data File Designation: "; Source$
PRINT "Data File Units: "; Unit$
PRINT "Data File Date: "; YMD&: PRINT
PRINT "Range: "; LowWav#; "-"; HiWav#; "nm @"; Intrvl#; "nm": PRINT
PRINT "Mechanical Range: "; GrtLow#; "-"; GrtHigh#; "nm"
PRINT

WvlnthEntry:
PRINT "Lower and upper limits must be wavelengths within the mechanical range."
PRINT : PRINT " ( Example: "; GrtLow#; ", "; GrtHigh#; ")": PRINT
PRINT Dashes$
PRINT "Enter lower & upper limits."
GOSUB BeepSound
INPUT LowWav#, HiWav#
GOSUB CheckLowHiWaves ' SUBROUTINE (3): Check LowWav# & HiWav#
CLS
PRINT "Measurement Range: "; LowWav#; "-"; HiWav#: PRINT

WvlnthJog:
PRINT : PRINT Dashes$
PRINT "Enter wavelength jog interval."
GOSUB BeepSound
INPUT Intrvl#
IF Intrvl# > 0 THEN CalcNumWvlnths
CLS
PRINT "The wavelength interval can not be <= 0 !": PRINT
GOTO WvlnthEntry

CalcNumWvlnths:
NumWvlnths% = (HiWav# - LowWav#) / Intrvl# + 1' # of scan points

DateEntry:
IF FLG1% = 1 THEN HdrAndDetctr' date entered & set up displayed
CLS

GetDate:
PRINT Dashes$
```

DRAFT

```
HdrData$ = DATE$ ' get default date
DateEntry$ = RIGHT$(HdrData$,2) + LEFT$(HdrData$,2) + MID$(HdrData$,4,2)
PRINT "Enter today's year, month, & day as: YMMDD (default = "; DateEntry$; ")."
GOSUB BeepSound
INPUT HdrData$ : IF HdrData$ <> "" THEN DateEntry$ = HdrData$
YMD& = VAL(DateEntry$)
IF YMD& = 0 OR YMD& > 999999! THEN GetDate
AVGFILE$ = "X"+DateEntry$+"@.AVG"
'
      Set default filename to Xymmdd@.AVG for now.
PRINT: PRINT

HardcopyPrompt:
PRINT "Output to printer (Y/N; Default = "; Hardcopy$; ")
INPUT A$
IF A$ = "" THEN A$ = Hardcopy$
IF A$ <> "Y" AND A$ <> "N" THEN HardcopyPrompt
Hardcopy$ = A$
CLS
PRINT "736 SET UP:"
PRINT " 1) chopper: 'ON'"
PRINT " 2) auto range: 'AUTO'"
PRINT " 3) response time: 'FAST'"
PRINT " 4) detector type: '1, 2, 3 or 4'"
PRINT " 5) phase angle: 'PRESET' or 'MANUAL': PRINT
PRINT "740-1C/D SET UP:"
PRINT " 1) monochromator wavelength counter: 'PRESET'"
PRINT " 2) filter wheel: 'COMPUTER'"
PRINT : PRINT Dashes$
WaitMessage$ = "Press space bar when SET UP is completed."
GOSUB SpaceToContinue
FLG1% = 1' date entered & set up displayed

HdrAndDetctr:
GOSUB XtraHdrData ' to SUBROUTINE (2) - EXTRA HEADER DATA
'
' - DETECTOR TYPE -
'
IF FLG2% = 1 THEN OpenAllfile' same det. type, noise level & designation
CLS
K1$ = "Type 1": K2$ = "Type 2": K3$ = "Type 3": K4$ = "Type 4"
K5$ = "": K6$ = "": K7$ = "": K8$ = "": K9$ = "": K10$ = ""
PRINT "Detector Type Option:": PRINT
PRINT "F1 : Si, PMT, Ge & InSb"
PRINT "F2 : PbS,PbSe"
PRINT "F3 : HgCdTe"
PRINT "F4 : Thermal Detector"
Q$ = "Select option."
GOSUB AssignKeyLabels ' SUBROUTINE (7): Assign key labels
ON FKey% GOTO DetectorType, DetectorType, DetectorType, DetectorType
GOTO HdrAndDetctr

DetectorType:
T = FKey%
Unit$ = "System output [Volts]"
IF T = 1 OR T = 4 THEN Unit$ = "System output [Amps]"
'
' - NOISE LEVEL -
'
K1$ = " 0.02%": K2$ = " 0.05%": K3$ = " 0.1% ": K4$ = " 0.5% "
K5$ = " 1% ": K6$ = " 2% ": K7$ = " 3% ": K8$ = " 4% "
K9$ = " 5% ": K10$ = " 10% "
CLS
PRINT "Detector Type: "; T: PRINT : PRINT
PRINT "Signal Noise Level Option:": PRINT
PRINT "F1 = very quiet...[0.02%]"
PRINT "F2 = .....[0.05%]"
PRINT "F3 = quiet.....[0.1%]"
PRINT "F4 = .....[0.5%]"
PRINT "F5 = noisy.....[1%]"
PRINT "F6 = .....[2%]"
PRINT "F7 = .....[3%]"
PRINT "F8 = very noisy...[4%]"
PRINT "F9 = .....[5%]"
```

DRAFT

```
PRINT "F10 = .....[10%]"
Q$ = "Select option."
GOSUB AssignKeyLabels ' SUBROUTINE (7): Assign key labels
IF FKey% = 1 THEN Noise# = .0002
IF FKey% = 2 THEN Noise# = .0005
IF FKey% = 3 THEN Noise# = .001
IF FKey% = 4 THEN Noise# = .005
IF FKey% = 5 THEN Noise# = .01
IF FKey% = 6 THEN Noise# = .02
IF FKey% = 7 THEN Noise# = .03
IF FKey% = 8 THEN Noise# = .04
IF FKey% = 9 THEN Noise# = .05
IF FKey% = 10 THEN Noise# = .1
CLS
PRINT "Signal Noise Level:"; 100 * Noise#; "%": PRINT

SrcCommentEntry:
PRINT "The run designation can be a descriptive name for the spectral data which
can"
PRINT "have a maximum length of 254 characters."
PRINT : PRINT "No commas (,) are allowed - use semi-colons (;) instead!"
ON ERROR GOTO CheckSrcComment ' SUBROUTINE (5): Check Source$
PRINT : PRINT Dashes$
PRINT "Enter run designation."
GOSUB BeepSound
INPUT Source$

OpenAllfile:
CLS
PRINT : PRINT Dashes$
PRINT : PRINT "Enter 746 output filename"
GOSUB FilenameEntry 'SUBROUTINE (0)
CLOSE #1: OPEN ALLFILE$ FOR OUTPUT AS #1
' *** header for 746 .ALL output file
PRINT #1, Source$ : PRINT #1, Unit$ : PRINT #1, "Date: ";YMD&
PRINT #1, USING "#####nm to #####nm #####nm steps ##### samples"; LowWav#; HiWav#;
Intrvl#; NumSmpls%
PRINT #1, USING "##c ##RH ##cm dist. #.##m slits"; RmTemp; RelHumid; SrcDst;
SlitSz
PRINT #1, "StartOfData:"
ON ERROR GOTO 0
'
' - MEASUREMENT ROUTINE -
'
OUT P1C, 144: FOR I = 1 TO 10 * MSM#: NEXT I
OUT P1C, 0 'Shutter/Neutral
' start time for delay to allow max time for shutter pos
FTIME1 = VAL(MID$(TIME$, 4, 2)) * 60 + VAL(MID$(TIME$, 7, 2))
IF NumGrtns% = 1 THEN GrtStType# = 1: GOTO OneGrating
FOR J% = 1 TO NumGrtns% - 1
IF LowWav# >= GrtngPrm#(J%, 3) AND LowWav# < GrtngPrm#(J% + 1, 3) THEN GrtStType#
= J%: GOTO OneGrating
NEXT J%
GrtStType# = NumGrtns%

OneGrating:
IF LowWav# < FltrWvlngth#(1) THEN FltStart% = 0: GOTO GratingPrompt
FOR J% = 1 TO NumFltrs% - 1
IF LowWav# >= FltrWvlngth#(J%) AND LowWav# < FltrWvlngth#(J% + 1) THEN
FltStart% = J%: GOTO GratingPrompt
NEXT J%

GratingPrompt:
CLS
PRINT "Place the "; GrtngPrm#(GrtStType#, 1); "um ("; GrtngPrm#(GrtStType#, 2); "
grooves/mm) grating in the optical beam path."
PRINT
PRINT "To abort run at any time press key 'F10'."
PRINT Dashes$
WaitMessage$ = "To start measurement press the space bar."
GOSUB SpaceToContinue
GOSUB BlankKeys
ON KEY(10) GOSUB AbortRun
```


DRAFT

```
TypeVolts:
    LPRINT " [nm]          [volts]          "

StartScan:
    ON ERROR GOTO 0
    FOR J% = 1 TO NumWvLngths%
    WavTest# = LowWav# + (J% - 1) * Intrvl#
    IF GrtCurrent# = NumGrtns% THEN SetWvlnthChkVars
    IF WavTest# < GrtngPrm#(GrtCurrent# + 1, 3) THEN SetWvlnthChkVars
    GrtCurrent# = GrtCurrent# + 1: WavForGrt# = 600 / GrtngPrm#(GrtCurrent#, 2)
    CLS
    PRINT "Place the "; GrtngPrm#(GrtCurrent#, 1); "um ("; GrtngPrm#(GrtCurrent#, 2);
" grooves/mm) grating in the optical beam path."
    PRINT : PRINT Dashes$
    WaitMessage$ = "Press the spacebar when grating is in place."
    GOSUB SpaceToContinue
    OUT P1C, 112: OUT P1C, 0 'Fast/Neutral
    OUT P1C, 192: OUT P1C, 224: OUT P1C, 0 'Reverse/Start/Neutral

GetNearWvlength:
    GOSUB ReadWavelength
    IF WaveRdng# > (WavTest# / WavForGrt#) - 10 THEN GetNearWvlength
    GOSUB ReadWavelength
    IF WaveRdng# > (WavTest# / WavForGrt#) - 10 THEN GetNearWvlength
    OUT P1C, 160: OUT P1C, 0 'Stop/Neutral

SetWvlnthChkVars:
    WvLength#(J%) = WavTest#: WavFactr# = WavTest# / WavForGrt#
    WaveSlow# = WavFactr# - .3: WaveChk# = WavFactr# - .4

ChkForFltrChange:
    IF FltCrrnt% = NumFltrs% THEN FineTuneWvlength
    IF WavTest# < FltrWvlnth#(FltCrrnt% + 1) THEN FineTuneWvlength
    OUT P1C, 176: FOR I = 1 TO 10 * MSM#: NEXT I: OUT P1C, 0
    'Advance/Neutral
    FOR I = 1 TO 1500 * MSM#: NEXT I ' Delay for filt position step
    FltCrrnt% = FltCrrnt% + 1: GOTO ChkForFltrChange

FineTuneWvlength:
    IF Intrvl# <= .3 AND J% > 1 THEN OUT P1C, 208 ELSE OUT P1C, 112: OUT P1C, 0
    ' slow/fast/neutral
    OUT P1C, 128: OUT P1C, 224: OUT P1C, 0' forward/start/neutral
    FOR I = 1 TO 10 * MSM#: NEXT I' delay loop

SlowlyDriveWvlength:
    WHILE WaveRdng# < WaveSlow#
    GOSUB ReadWavelength ' SUBROUTINE (9): Read wavelength
    WEND
    GOSUB ReadWavelength ' SUBROUTINE (9): Read wavelength
    IF WaveRdng# < WaveChk# THEN SlowlyDriveWvlength
    OUT P1C, 208: OUT P1C, 0' slow/neutral
    WHILE WaveRdng# < WavFactr#
    GOSUB ReadWavelength ' SUBROUTINE (9): Read wavelength
    WEND
    OUT P1C, 160: OUT P1C, 0' stop/neutral
    GOSUB Read736Signal: GOSUB ReadWavelength ' read signal & wavelength
    PRINT USING " #####.##nm    (#####.##)"; WavTest#; WavFactr#

BeginSampling:
    SumSgnls# = 0: SumSqrSgnls# = 0: TimeMrk$ = TIME$ ' get time mark
    PRINT #1, WavTest#; " "; TimeMrk$; ' print time & wavelength to .ALL file

    FOR Sampl% = 1 TO NumSmpls%
        GOSUB ShowEachSignal ' SUBROUTINE (14)
    NEXT Sampl%

    SgnlAvg# = (SumSgnls# / NumSmpls%)
    CheckSqrSum# = NumSmpls% * SumSqrSgnls# - SumSgnls# ^ 2
    IF CheckSqrSum# < 0 then CheckSqrSum# = 0
    SgnlPctDev# = SQR(CheckSqrSum# / (NumSmpls% * (NumSmpls% - 1))) / SgnlAvg#

    FOR Sampl% = NumSmpls% + 1 TO MaxSmpls%
        IF SgnlPctDev# <= Noise# THEN SkipTheRest
```

DRAFT

```
GOSUB ShowEachSignal ' SUBROUTINE (14)
SgnlAvg# = (SumSgnls# / Sampl%)
CheckSqrSum# = Sampl% * SumSqrSgnls# - SumSgnls# ^ 2
IF CheckSqrSum# < 0 then CheckSqrSum# = 0
SgnlPctDev# = SQR(CheckSqrSum# / (Sampl% * (Sampl% - 1))) / SgnlAvg#
SkipTheRest:
NEXT Sampl%

PRINT #1, ""
PRINT : PRINT USING "          Average = ##.###^" SgnlAvg#
PRINT USING "Std. Dev. / Avg. = ##.###" SgnlPctDev# * 100: PRINT
AvgSgnl#(J%) = SgnlAvg#
IF Hardcopy$ = "N" THEN GetNextWavelength
'
' - PRINT DATA ROUTINE -
'
ON ERROR GOTO CheckPrinter ' SUBROUTINE (10): Error-check printer
IF H% - 5 = 0 THEN PrintWvlAndSgnl
LPRINT USING "          ##.###^"          ##.###^"; AvgSgnl#(J%); SgnlPctDev#
H% = H% + 1' counter, printout format
GOTO GetNextWavelength

PrintWvlAndSgnl:
LPRINT
LPRINT USING "#####.##          ##.###^"          ##.###^"; WavTest#; AvgSgnl#(J%);
SgnlPctDev#
H% = 1' counter, printout format

GetNextWavelength:
NEXT J%
CLOSE #1
FLG3% = 0' scan complete
IF Hardcopy$ = "N" THEN SelectOption
LPRINT : LPRINT Dashes$
LPRINT : LPRINT : LPRINT : LPRINT ' end scan

SelectOption:
ON ERROR GOTO 0
CLS
IF FLG3% = 0 THEN KeysForCompletedRun ELSE KeysForAbortedRun

KeysForCompletedRun:
K1$ = " Plot ": K2$ = "": K3$ = " Store": K4$ = "": K5$ = "Same R"
K6$ = "": K7$ = " New R": K8$ = "": K9$ = "": K10$ = " End "
GOTO WholeMenu

KeysForAbortedRun:
K1$ = "": K2$ = "": K3$ = "": K4$ = "": K5$ = "Same R"
K6$ = "": K7$ = " New R": K8$ = "": K9$ = "": K10$ = " End "

WholeMenu:
PRINT "Current Calibration Range:"; LowWav#; "-"; HiWav#; "@"; Intrvl#; "nm":
PRINT
IF FLG3% = 1 THEN RestOfMenu' scan aborted
PRINT "F1 : plot signal output data"
PRINT "F3 : store signal output data"

RestOfMenu:
PRINT "F5 : scan over same range & interval"
PRINT "F7 : scan over new range &/or interval"
PRINT "F10: end"
Q$ = "Select option."
GOSUB AssignKeyLabels ' SUBROUTINE (7): Assign key labels

EOSFncKey:
ON FKey% GOTO PlotScan, EOSFncKey, StoreAvgData, EOSFncKey, DoAnotherScan,
EOSFncKey, DoAnotherScan, EOSFncKey, EOSFncKey, ExitProgram

DoAnotherScan:
Tmp1% = FKey%
'
' - CHANGE OPTION -
'
```

DRAFT

```
CLS
K1$ = "change": K2$ = "": K3$ = "": K4$ = " "
K5$ = "": K6$ = "": K7$ = "": K8$ = "": K9$ = "": K10$ = " same"
PRINT "Run Designation: "; Source$
PRINT "Signal Noise Level: "; Noise# * 100; "%"
PRINT "Detector Type: "; T: PRINT
PRINT "F1 : change detector type, noise level &/or designation"
PRINT "F10 : use same detector type, noise level & designation"
Q$ = "Select option."
GOSUB AssignKeyLabels ' SUBROUTINE (7): Assign key labels

ChangeOp:
  ON FKey% GOTO OpKeyHit, ChangeOp, ChangeOp, ChangeOp, ChangeOp, ChangeOp,
ChangeOp, ChangeOp, ChangeOp, OpKeyHit

OpKeyHit:
  FLG2% = 0: IF FKey% = 10 THEN FLG2% = 1
  GOTO GRFileLoaded ' measurement parameters

StoreAvgData:
  ' - DATA STORAGE ROUTINE -
  CLS
  ON ERROR GOTO FileCreationError ' SUBROUTINE (4)
  CLOSE #1: OPEN AVGFILE$ FOR OUTPUT AS #1
  WRITE #1, Source$, Unit$, YMD&, LowWav#, HiWav#, Intrvl#
  FOR I = 1 TO NumWvLnths%
  PRINT #1, USING "##.#####^"^"; AvgSgnl#(I)
  NEXT I
  CLOSE #1
  ON ERROR GOTO 0
  IF Hardcopy$ = "N" THEN SelectOption
  ON ERROR GOTO CheckPrinter ' SUBROUTINE (10)
  LPRINT Dashes$
  LPRINT "AVERAGE SYSTEM SIGNAL OUTPUT DATA FILE"
  LPRINT : LPRINT "Saved as: "; AVGFILE$
  LPRINT "Designation: "; Source$
  LPRINT "Units: "; Unit$
  LPRINT "Date: "; YMD&
  LPRINT "Lower = "; LowWav#
  LPRINT "Upper = "; HiWav#
  LPRINT "Interval = "; Intrvl#
  LPRINT Dashes$
  LPRINT : LPRINT : LPRINT : LPRINT
  ON ERROR GOTO 0
  GOTO SelectOption

AbortRun:
  ' - ABORT ROUTINE -
  '
  OUT P1C, 160: OUT P1C, 0' stop/neutral
  IF Hardcopy$ = "N" THEN ShowAbort
  ON ERROR GOTO CheckPrinter ' SUBROUTINE (10): Error-check printer
  LPRINT : LPRINT "----- ABORT -----"
  LPRINT Dashes$: LPRINT : LPRINT : LPRINT : LPRINT

ShowAbort:
  ON ERROR GOTO 0
  FLG3% = 1: RETURN SelectOption' scan aborted

ExitProgram:
  ' - END OF PROGRAM -
  '
  CLS : PRINT "End of program!"
  SOUND 500, 5 ' signals end of program
  END

  ' ----- 0 -----

FilenameEntry:
  PRINT "The data file name can be 1-8 characters in length. Valid characters
include:"
  PRINT "A-Z 0-9 ( ) { } @ # $ % ^ & ! - _ ` ' / ~"
  PRINT
  PRINT "Any other characters are invalid. The program will "
```

DRAFT

```
PRINT "automatically add the extensions .ALL and .AVG "  
PRINT "to designate data files."  
' strip out .extension  
AVGFILE$ = LEFT$(AVGFILE$, 8): LocDot% = INSTR(1, AVGFILE$, ".")  
IF LocDot% <> 0 THEN AVGFILE$ = LEFT$(AVGFILE$, LocDot% - 1)  
' add alpha postscript for sequencing  
AVGFILE$ = LEFT$(AVGFILE$, LEN(AVGFILE$)-1) + CHR$(ASC(RIGHT$(AVGFILE$,1)) + 1)  
PRINT : PRINT Dashes$  
PRINT "Enter name of data file where values are to be stored."  
PRINT "(Default = "; AVGFILE$ ;")"  
GOSUB BeepSound  
INPUT HdrData$  
IF HdrData$<>" THEN AVGFILE$ = HdrData$  
AVGFILE$ = LEFT$(AVGFILE$, 8): LocDot% = INSTR(1, AVGFILE$, ".")  
IF LocDot% <> 0 THEN AVGFILE$ = LEFT$(AVGFILE$, LocDot% - 1)  
ALLFILE$ = AVGFILE$ + ".ALL": AVGFILE$ = AVGFILE$ + ".AVG"  
IF AVGFILE$ = ".AVG" THEN FilenameEntry  
RETURN  
  
' ----- 1 -----  
PlotScan:  
' SUBROUTINE (1): Graphics display  
  
' - data sorting routine -  
'  
MaxSgnl# = AvgSgnl#(1)' maximum  
FOR I = 1 TO NumWvLngths%  
IF AvgSgnl#(I) > MaxSgnl# THEN MaxSgnl# = AvgSgnl#(I)  
NEXT I  
  
' - normalization loop -  
'  
FOR I = 1 TO NumWvLngths%  
IF AvgSgnl#(I) = MaxSgnl# THEN WavTest# = WvLength#(I)  
NrmVal#(I) = AvgSgnl#(I) / MaxSgnl#  
NEXT I  
  
' - draw grid -  
'  
SCREEN 2: CLS : KEY OFF  
  
' - X-axis lines (horizontal) -  
'  
FOR Y = 3 TO 163 STEP 16  
LINE (30, Y)-(610, Y)  
NEXT Y  
  
' - Y labels -  
'  
ROW = 21  
FOR YLABEL = 0 TO 1.1 STEP .1  
LOCATE ROW, 1  
PRINT USING "#.#"; YLABEL  
ROW = ROW - 2  
NEXT YLABEL  
  
' - Y-axis lines (vertical) & X labels -  
'  
YAxisInc% = 1' Y-axis increment  
  
GetYLines:  
IF (HiWav# - LowWav#) / (Intrvl# * YAxisInc%) <= 10 THEN GetXLabels  
YAxisInc% = YAxisInc% + 1: GOTO GetYLines  
  
GetXLabels:  
FOR X = 0 TO (HiWav# - LowWav#) / (Intrvl# * YAxisInc%)  
WaveLabel# = LowWav# + Intrvl# * YAxisInc% * X  
WaveGridPos# = (WaveLabel# - LowWav#) * (610 - 30) / (HiWav# - LowWav#) + 30  
LINE (WaveGridPos#, 163)-(WaveGridPos#, 3)  
WaveLabelPos# = (WaveLabel# - LowWav#) * .125 * (610 - 30) / (HiWav# - LowWav#) +  
.125 * 30  
IF WaveLabel# >= 1000 THEN LOCATE 22, WaveLabelPos# - 1 ELSE LOCATE 22,  
WaveLabelPos#
```

DRAFT

```
IF WaveLabel# >= 1000 THEN PRINT USING "#####"; WaveLabel# ELSE PRINT USING "###";
WaveLabel#
NEXT X

' - plot data -
'
PSET (30, (1 - NrmVal#(1)) * (163 - 3) / (1 - 0) + 3)
FOR I = 1 TO NumWvLngths%
LINE -((WvLength#(I) - LowWav#) * (610 - 30) / (HiWav# - LowWav#) + 30, (1 -
NrmVal#(I)) * (163 - 3) / (1 - 0) + 3)
NEXT I
LOCATE 23, 20: PRINT USING "Peak Value = ###.###^ ^ ^ @ #####. #nm"; MaxSgnl#;
WavTest#
IF Hardcopy$ = "N" THEN LookAtGraph

' - dump graphics display to printer -
'
ON ERROR GOTO CheckPrinter ' SUBROUTINE (10)
DEF SEG = &H40
HIGHMEM = (((256 * PEEK(20) + PEEK(19)) / 64) - 1) * 4096 + 3988
DEF SEG = HIGHMEM
SBRPTSC = 0
POKE 0, 205: POKE 1, 5: POKE 2, 203
DEF SEG = HIGHMEM: CALL ABSOLUTE SBRPTSC

' - print data file heading -
'
LPRINT
LPRINT "Data File Designation: "; Source$
LPRINT "Data File Units: "; Unit$
LPRINT "Date File Date: "; YMD&
LPRINT : LPRINT USING "Wavelength Range: #####.# to #####.#nm @ ###.#nm";
LowWav#; HiWav#; Intrvl#
LPRINT : LPRINT USING "Peak Value = ###.###^ ^ ^ @ #####. #nm"; MaxSgnl#; WavTest#
LPRINT : LPRINT : LPRINT : LPRINT
LPRINT : LPRINT : LPRINT : LPRINT
ON ERROR GOTO 0
SCREEN 0: GOTO SelectOption ' return to 'OPTION' menu

LookAtGraph:
LOCATE 23, 1
WaitMessage$ = "SPC TO CONT."
GOSUB SpaceToContinue
SCREEN 0: GOTO SelectOption ' return to 'OPTION' menu

' ----- 2 -----
XtraHdrData:
CLS ' SUBROUTINE (2): additional header data
PRINT Dashes$
PRINT "Enter lab temperature (";RmTemp;"c)"
GOSUB BeepSound
INPUT HdrData$: IF HdrData$ <> "" THEN RmTemp = VAL(HdrData$)
PRINT "Enter relative humidity (";RelHumid;"%)"
GOSUB BeepSound
INPUT HdrData$: IF HdrData$ <> "" THEN RelHumid = VAL(HdrData$)
PRINT "Enter distance to source (";SrcDst;"cm)"
GOSUB BeepSound
INPUT HdrData$: IF HdrData$ <> "" THEN SrcDst = VAL(HdrData$)
PRINT "Enter slit size (";SlitSz;"mm); enter 0 for special"
GOSUB BeepSound
INPUT HdrData$: IF HdrData$ <> "" THEN SlitSz = VAL(HdrData$)

EnterNumSamples:
PRINT
PRINT "Enter minimum number of samples to be taken per wavelength (";NumSmpls%;"")"
GOSUB BeepSound
INPUT HdrData$: IF HdrData$ <> "" THEN NumSmpls% = VAL(HdrData$)
IF NumSmpls% > 100 OR NumSmpls% <= 1 THEN PRINT "MUST BE <= 100 AND > 1": GOTO
EnterNumSamples

PRINT
PRINT "Enter maximum number of samples to be taken per wavelength "
PRINT "if the noise level is not achieved (";MaxSmpls%;"")"
```

DRAFT

```
GOSUB BeepSound
INPUT HdrData$: IF HdrData$ <> "" THEN MaxSmpls% = VAL(HdrData$)
IF MaxSmpls% > 100 OR MaxSmpls% < NumSmpls% THEN PRINT "MUST BE <= 100 AND > ";
NumSmpls%: GOTO EnterNumSamples
RETURN

' ----- 3 -----
CheckLowHiWaves:
' SUBROUTINE (3): Check LowWav# & HiWav#
CLS
IF LowWav# < HiWav# THEN CheckLowMechRange
PRINT LowWav#; "is not <"; HiWav#; "!": PRINT
PRINT "The lower wavelength limit must be < the upper wavelength limit!"
PRINT : PRINT : RETURN WvlngtEntry

CheckLowMechRange:
IF LowWav# >= GrtLow# THEN CheckHiMechRange
PRINT LowWav#; "is < the lower mechanical limit of"; GrtLow#; "nm": PRINT : RETURN
WvlngtEntry

CheckHiMechRange:
IF HiWav# <= GrtHigh# THEN RETURN
PRINT HiWav#; "is > the upper mechanical limit of"; GrtHigh#; "nm": PRINT : RETURN
WvlngtEntry
RETURN

' ----- 4 -----
FileCreationError:
' SUBROUTINE (4): Error creating AVGFILE$
'
IF ERR = 70 THEN CLS ELSE ChkDiskFull
PRINT "Mass storage medium is write-protected!"
PRINT : PRINT Dashes$
WaitMessage$ = "Press space bar when fixed."

WaitForDisk:
GOSUB SpaceToContinue
RESUME StoreAvgData ' return to OPEN AVGFILE$

ChkDiskFull:
IF ERR = 61 THEN CLS ELSE ChkFileSpec
PRINT "All diskette storage space is in use!"
PRINT : PRINT "If there are any files on the diskette that you no longer need, "
PRINT "erase them or use a new diskette. Then retry the operation."
PRINT : PRINT Dashes$
WaitMessage$ = "Press space bar when ready."
GOTO WaitForDisk

ChkFileSpec:
IF ERR = 67 THEN CLS ELSE ChkDiskDoor
PRINT "The file directory on the mass storage medium is full "
PRINT "or the file specification is invalid!"
PRINT : PRINT Dashes$
PRINT "If the file specification is okay, use a new formatted disc & retry
operation."
WaitMessage$ = "Press space bar when ready."
GOTO WaitForDisk

ChkDiskDoor:
IF ERR = 71 THEN CLS ELSE UnIDError
PRINT "The disc door is open or a disc is not in the drive.": PRINT
PRINT "Place the correct disc in the default drive."
PRINT : PRINT Dashes$
WaitMessage$ = "Press space bar when ready."
GOTO WaitForDisk

UnIDError:
GOSUB AlertUserOfError ' SUBROUTINE (6): Unidentified error
RESUME StoreAvgData ' return to DATA STORAGE ROUTINE

' ----- 5 -----
CheckSrcComment:
' SUBROUTINE (5): Error-check Source$
```

DRAFT

```
'
IF ERRN <> 15 THEN UnknownStringError
PRINT
PRINT "The length of this input is > 254 characters!"
RESUME SrcCommentEntry ' return to enter designation

UnknownStringError:
  GOSUB AlertUserOfError ' SUBROUTINE (6): Unidentified error
  RESUME SrcCommentEntry ' return to enter designation

' ----- 6 -----
AlertUserOfError:
  ' SUBROUTINE (6): Unidentified error
  '
  CLS
  PRINT "Error #"; ERR; "on line"; ERL; "!": PRINT
  PRINT "Please refer to your TURBO BASIC manual."
  PRINT : PRINT Dashes$
  WaitMessage$ = "Press space bar when ready."
  GOSUB SpaceToContinue
  RETURN

' ----- 7 -----
AssignKeyLabels:
  ' SUBROUTINE (7)
  FKey% = 0
  PRINT : PRINT "Use 'soft key' answer."
  PRINT Dashes$
  PRINT Q$
  KEY 1, K1$: KEY 2, K2$: KEY 3, K3$: KEY 4, K4$: KEY 5, K5$
  KEY 6, K6$: KEY 7, K7$: KEY 8, K8$: KEY 9, K9$: KEY 10, K10$
  SOUND 500, 1
  IF K1$ = "" THEN ON KEY(1) GOSUB NoMore ELSE ON KEY(1) GOSUB F1
  IF K2$ = "" THEN ON KEY(2) GOSUB NoMore ELSE ON KEY(2) GOSUB F2
  IF K3$ = "" THEN ON KEY(3) GOSUB NoMore ELSE ON KEY(3) GOSUB F3
  IF K4$ = "" THEN ON KEY(4) GOSUB NoMore ELSE ON KEY(4) GOSUB F4
  IF K5$ = "" THEN ON KEY(5) GOSUB NoMore ELSE ON KEY(5) GOSUB F5
  IF K6$ = "" THEN ON KEY(6) GOSUB NoMore ELSE ON KEY(6) GOSUB F6
  IF K7$ = "" THEN ON KEY(7) GOSUB NoMore ELSE ON KEY(7) GOSUB F7
  IF K8$ = "" THEN ON KEY(8) GOSUB NoMore ELSE ON KEY(8) GOSUB F8
  IF K9$ = "" THEN ON KEY(9) GOSUB NoMore ELSE ON KEY(9) GOSUB F9
  IF K10$ = "" THEN ON KEY(10) GOSUB NoMore ELSE ON KEY(10) GOSUB F10
  KEY(1) ON: KEY(2) ON: KEY(3) ON: KEY(4) ON: KEY(5) ON
  KEY(6) ON: KEY(7) ON: KEY(8) ON: KEY(9) ON: KEY(10) ON
  KEY ON

F0:
  IF FKey% = 0 THEN F0 ELSE KEY OFF: RETURN

F1:
  FKey% = 1: GOTO KeysOff

F2:
  FKey% = 2: GOTO KeysOff

F3:
  FKey% = 3: GOTO KeysOff

F4:
  FKey% = 4: GOTO KeysOff

F5:
  FKey% = 5: GOTO KeysOff

F6:
  FKey% = 6: GOTO KeysOff

F7:
  FKey% = 7: GOTO KeysOff

F8:
  FKey% = 8: GOTO KeysOff

F9:
  FKey% = 9: GOTO KeysOff

F10:
  FKey% = 10

KeysOff:
  KEY(1) OFF: KEY(2) OFF: KEY(3) OFF: KEY(4) OFF: KEY(5) OFF
  KEY(6) OFF: KEY(7) OFF: KEY(8) OFF: KEY(9) OFF: KEY(10) OFF
  GOSUB BlankKeys
```

DRAFT

```
K1$ = "": K2$ = "": K3$ = "": K4$ = "": K5$ = ""
K6$ = "": K7$ = "": K8$ = "": K9$ = "": K10$ = ""
RETURN

NoMore:
  IF INKEY$ = "T" THEN BeepTenSeconds ELSE SOUND 500, 1: RETURN

BeepTenSeconds:
  ' 10 sec test
  BEEP: FOR J0% = 1 TO 10000 * MSM#: NEXT J0%: BEEP: RETURN

  ' ----- 8 -----
Read736Signal:
  ' SUBROUTINE (8)
  '
  FOR I = 1 TO 450 * MSM#: NEXT I' delay loop

GetPortContents:
  PORT2A = INP(P2A): PORT2B = INP(P2B): PORT2C = INP(P2C)
  DATAREADY = ((PORT2C AND 8) / 8)
  IF DATAREADY = 1 THEN GetPortContents
  PORT2AHIGH = (PORT2A AND HIMASK) / 16: PORT2ALO = (PORT2A AND LOMASK)
  PORT2BHIGH = (PORT2B AND HIMASK) / 16: PORT2BLO = (PORT2B AND LOMASK)
  PORT2C1 = PORT2C AND 1
  EXPONENT = ((PORT2C AND 4) / 4)
  POWER = -(EXPONENT * 10 + PORT2ALO)
  MANTISSA# = CDBL(PORT2C1 * 1 + PORT2BHIGH * .1 + PORT2BLO * .01 + PORT2AHIGH *
.001)
  Sgn1Out# = CDBL(MANTISSA# * (10 ^ POWER)): RETURN

  ' ----- 9 -----
ReadWavelength:
  ' SUBROUTINE (9): Read 740-1C wavelength
  '
  PORT1A = INP(P1A): PORT1B = INP(P1B): PORT1C = INP(P1C)
  PORT1AHIGH = (PORT1A AND HIMASK) / 16: PORT1ALO = PORT1A AND LOMASK
  PORT1BHIGH = (PORT1B AND HIMASK) / 16: PORT1BLO = PORT1B AND LOMASK
  PORT1C3 = PORT1C AND 3
  WaveRdng# = CDBL(PORT1C3 * 1000 + PORT1BHIGH * 100 + PORT1BLO * 10 + PORT1AHIGH *
1 + PORT1ALO * .1)
  RETURN

  ' ----- 10 -----
CheckPrinter:
  ' SUBROUTINE (10): Error-check printer
  '
  IF ERR = 24 THEN CLS ELSE PrinterOffLine
  PRINT "Device timeout!": PRINT
  PRINT "The printer is not online, or the cable connector is loose or faulty.":
PRINT
  PRINT Dashes$
  WaitMessage$ = "Press space bar when fixed."
  GOTO WaitForPrintFix

PrinterOffLine:
  IF ERR = 25 THEN CLS ELSE OutOfPaper
  PRINT "Device fault!": PRINT
  PRINT "Printer is off or not online.": PRINT
  PRINT Dashes$
  WaitMessage$ = "Press space bar when printer is ready."
  GOTO WaitForPrintFix

OutOfPaper:
  IF ERR = 27 THEN CLS ELSE UnknownPrintError
  PRINT "The printer is out of paper, or not switched on."
  PRINT : PRINT Dashes$
  WaitMessage$ = "Press space bar when fixed."

WaitForPrintFix:
  GOSUB SpaceToContinue
  RESUME

UnknownPrintError:
```

DRAFT

```
GOSUB AlertUserOfError ' SUBROUTINE (6): Unidentified error
RESUME

'-----11-----
SpaceToContinue:
  PRINT WaitMessage$
  GOSUB BeepSound

WaitForSpace:
  A$ = INKEY$: IF A$ <> " " THEN WaitForSpace
  RETURN

'-----12-----
BeepSound:
  SOUND 500, 1: DEF SEG = 0: POKE 1050, PEEK(1052)
  RETURN

'-----13-----
BlankKeys:
  KEY 1, "": KEY 2, "": KEY 3, "": KEY 4, "": KEY 5, "
  KEY 6, "": KEY 7, "": KEY 8, "": KEY 9, "
  RETURN

'-----14-----
ShowEachSignal:
  GOSUB Read736Signal: Sgnl#(Sampl%) = SgnlOut# ' get signal number Sampl%
  SumSgnls# = SumSgnls# + Sgnl#(Sampl%)
  SumSqrSgnls# = SumSqrSgnls# + Sgnl#(Sampl%) ^ 2
  PRINT #1, USING " ###.###^^^"; Sgnl#(Sampl%); ' output signal data to .ALL file
  PRINT USING " ###.###^^^"; Sgnl#(Sampl%);
  RETURN

LoadFltGrtFile:
  ' SUBPROGRAM: GR-FL
  ' DATE: 6/3/87
  '
  ' DESCRIPTION: Subprogram for development of grating selection routine
  '               and filter control for insertion into 746 measurement
  '               programs to accomodate 740-1C/D Controller and G-1000 Turret
  '               Grating mount.
  '
  ' DEFINITION OF VARIABLES
  ' =====
  '
  ' FltrWvlngth#(I)   : Crossover wavelengths for each filter position
  ' GrtngPrm#(I,J%)  : Grating Information for -
  '                   position I (up to 3), parameter J%
  '                   where  J%=1 ; Blaze wavelength
  '                   J%=2 ; Grooves per millimeter
  '                   J%=3 ; Wavelength crossover
  '
  FR = 0
  '
  '       READ "GR-FL.DAT" CONFIGURATION FILE
  '       =====
  ON ERROR GOTO ChkForGratingFile
  CLOSE #1: OPEN "GR-FL.DAT" FOR INPUT AS #1
  ON ERROR GOTO 0
  ON ERROR GOTO UnknownGratingFileError
  INPUT #1, NumFltrs%
  FOR I = 1 TO NumFltrs%
    INPUT #1, FltrWvlngth#(I)
  NEXT I
  INPUT #1, NumGrtngs%
  FOR I = 1 TO NumGrtngs%
    INPUT #1, GrtngPrm#(I, 1), GrtngPrm#(I, 2), GrtngPrm#(I, 3)
```

DRAFT

```
        NEXT I
    CLOSE #1
    ON ERROR GOTO 0
    GOTO DisplayGratingDefaults

ChkForGratingFile:
    IF ERR = 53 THEN CLS ELSE UnknownGratingFileError
        PRINT "Filter and Grating data file (GR-FL.DAT) not present."
        PRINT "Place disk with file in disk drive."
        PRINT : PRINT Dashes$
    WaitMessage$ = "Press the spacebar when ready."
    GOSUB SpaceToContinue
    RESUME

UnknownGratingFileError:
    GOSUB AlertUserOfError 'Unidentified ERROR
    RESUME

DisplayGratingDefaults:
    '        DISPLAY FILTER AND GRATING DEFAULT VALUES
    '        =====
    '
    FIMG$ = "                Position ##: #####nm"
    GIMG$ = "        Grating #:  blaze:  ##.##um  ##### grooves/mm  cut on:  #####nm"
    '
    CLS
    PRINT : PRINT "Filter cut-on wavelengths.": PRINT
    FOR I = 1 TO NumFltrs%
    PRINT USING FIMG$; I; FltrWvlngth#(I)
    NEXT I
    PRINT : PRINT "Grating types selected.": PRINT
    FOR I = 1 TO NumGrtns%
    PRINT USING GIMG$; I; GrtnGPrm#(I, 1); GrtnGPrm#(I, 2); GrtnGPrm#(I, 3)
    NEXT I
    Q$ = "Select option desired."
    K1$ = "Filter": K3$ = "Grating": K5$ = "Store": K10$ = "Continue"
    K2$ = "": K4$ = "": K6$ = "": K7$ = "": K8$ = "": K9$ = ""
    GOSUB AssignKeyLabels

GrtnOps:
    ON FKey% GOTO NewFltr, GrtnOps, NewGrtn, GrtnOps, StoreFlGrtn, GrtnOps, GrtnOps, GrtnOps,
    GrtnOps, ContGrProgram

ContGrProgram:
    IF GrtnGPrm#(NumGrtns%, 1) > 99 THEN ShowGratingTypes
    GrtnLow# = GrtnGPrm#(1, 3): GrtnHigh# = 2500 * (600 / GrtnGPrm#(NumGrtns%, 2))
    GOTO GRFileLoaded

ShowGratingTypes:
    CLS : PRINT "Grating types selected": PRINT
    FOR I = 1 TO NumGrtns%
    PRINT USING GIMG$; I; GrtnGPrm#(I, 1); GrtnGPrm#(I, 2); GrtnGPrm#(I, 3)
    NEXT I: PRINT
    PRINT "Grating # "; NumGrtns%; " has not been defined or the parameters are not
valid for this"
    PRINT "measurement system. Please redefine the parameters for grating # ";
NumGrtns%; " or"
    PRINT "delete it from the gratings to be used in the measurement program."
    GOTO GratingMenu ' grating selection menu

NewFltr:
    '        FILTER CHANGE
    CLS
    PRINT "Filter cut-on wavelengths for each position": PRINT
    FOR I = 1 TO NumFltrs%
    PRINT USING FIMG$; I; FltrWvlngth#(I)
    NEXT I
    PRINT : PRINT Dashes$: PRINT
    PRINT "F1 : Select position to be changed."
    PRINT "F10: Exit CHANGE FILTER menu."
    Q$ = "Select option desired."
    K1$ = "Select": K10$ = "Exit"
    K2$ = "": K3$ = "": K4$ = "": K5$ = ""
```

DRAFT

```
K6$ = "": K7$ = "": K8$ = "": K9$ = ""
GOSUB AssignKeyLabels

FltrOps:
    ON FKey% GOTO SelectFilter, FltOps, FltOps, FltOps, FltOps, FltOps, FltOps,
    FltOps, FltOps, DisplayGratingDefaults

SelectFilter:
    DEF SEG = 0: POKE 1050, PEEK(1052)
    LOCATE 22, 1: BEEP: INPUT "Enter position to be changed? ", POST
    IF POST < 1 OR POST > NumFltrs% THEN BEEP: GOTO SelectFilter
    LOCATE 2 + POST, 31: PRINT SPACE$(5)
    DEF SEG = 0: POKE 1050, PEEK(1052)
    LOCATE 2 + POST, 29: BEEP: INPUT CHANGE
    IF POST <> 1 THEN NotFirstFilter
    IF CHANGE > FltrWvlngth#(POST + 1) THEN FltrNotInSequence ELSE MkFilterChange

NotFirstFilter:
    IF POST <> NumFltrs% THEN NotLastFilter
    IF CHANGE < FltrWvlngth#(POST - 1) THEN FltrNotInSequence ELSE MkFilterChange

NotLastFilter:
    IF CHANGE <= FltrWvlngth#(POST + 1) OR CHANGE >= FltrWvlngth#(POST - 1) THEN
    MkFilterChange

FltrNotInSequence:
    CLS : BEEP
    PRINT "Wavelength entered not in sequence with other filters."
    PRINT : PRINT Dashes$: WaitMessage$ = "Press spacebar to continue."
    GOSUB SpaceToContinue
    GOTO NewFltr

MkFilterChange:
    FltrWvlngth#(POST) = CHANGE: GOTO NewFltr

NewGrt:
    '          GRATING CHANGE
    CLS
    PRINT "Grating types selected": PRINT
    FOR I = 1 TO NumGrtns%
    PRINT USING GIMG$: I; GrtngPrm#(I, 1); GrtngPrm#(I, 2); GrtngPrm#(I, 3)
    NEXT I

GratingMenu:
    PRINT : PRINT Dashes$: PRINT
    PRINT "F1 : Change number of gratings."
    PRINT "F3 : Select position to be changed."
    PRINT "F10: Exit to CHANGE menu."
    Q$ = "Select option desired."
    K1$ = "Number": K3$ = "Select": K10$ = "Exit"
    K2$ = "": K4$ = "": K5$ = "": K6$ = ""
    K7$ = "": K8$ = "": K9$ = ""
    GOSUB AssignKeyLabels

FltrOps:
    ON FKey% GOTO EnterNumOfGrts, FltrOps, SelectGrating, FltrOps, FltrOps, FltrOps,
    FltrOps, FltrOps, FltrOps, DisplayGratingDefaults

EnterNumOfGrts:
    LOCATE 21, 1: BEEP
    DEF SEG = 0: POKE 1050, PEEK(1052)
    INPUT "Enter number of gratings to be defined (up to 3)? ", CHANGE
    IF CHANGE < 1 OR CHANGE > 3 THEN EnterNumOfGrts
    IF CHANGE <= NumGrtns% THEN FewerGratings
    FOR I = NumGrtns% + 1 TO CHANGE
    GrtngPrm#(I, 1) = 99.99: GrtngPrm#(I, 2) = 99999!
    GrtngPrm#(I, 3) = 99999!
    NEXT I

FewerGratings:
    IF CHANGE >= NumGrtns% THEN MoreGratings
    FOR I = CHANGE + 1 TO 3
    GrtngPrm#(I, 1) = 99.99: GrtngPrm#(I, 2) = 99999!
```

DRAFT

```
GrtnGPrm#(I, 3) = 99999!
NEXT I

MoreGratings:
  NumGrtnG% = CHANGE: GOTO NewGrt

SelectGrating:
  DEF SEG = 0: POKE 1050, PEEK(1052)
  LOCATE 21, 1: BEEP: INPUT "Enter position to be changed? ", POST
  IF POST < 1 OR POST > NumGrtnG% THEN BEEP: GOTO SelectGrating
  LOCATE 2 + POST, 27: PRINT SPACE$(5)
  LOCATE 2 + POST, 37: PRINT SPACE$(5)
  LOCATE 2 + POST, 65: PRINT SPACE$(5)
  DEF SEG = 0: POKE 1050, PEEK(1052)
  LOCATE 2 + POST, 25: BEEP: INPUT CHANGE1
  DEF SEG = 0: POKE 1050, PEEK(1052)
  LOCATE 2 + POST, 35: BEEP: INPUT CHANGE2
  DEF SEG = 0: POKE 1050, PEEK(1052)
  LOCATE 2 + POST, 63: BEEP: INPUT CHANGE3
  IF POST <> 1 THEN PostNot1
  IF CHANGE1 > GrtnGPrm#(POST + 1, 1) THEN GrtNotInSeq
  IF CHANGE3 > GrtnGPrm#(POST + 1, 3) THEN WaveNotInSeq ELSE ChangeGrtParams

PostNot1:
  IF POST <> NumGrtnG% THEN PostNotNumGrts
  IF CHANGE1 < GrtnGPrm#(POST - 1, 1) THEN GrtNotInSeq
  IF CHANGE3 < GrtnGPrm#(POST - 1, 3) THEN WaveNotInSeq ELSE ChangeGrtParams

PostNotNumGrts:
  IF CHANGE1 > GrtnGPrm#(POST + 1, 1) OR CHANGE1 < GrtnGPrm#(POST - 1, 1) THEN
GrtNotInSeq
  IF CHANGE3 > GrtnGPrm#(POST + 1, 3) OR CHANGE3 < GrtnGPrm#(POST - 1, 3) THEN
WaveNotInSeq ELSE ChangeGrtParams

GrtNotInSeq:
  CLS : BEEP
  PRINT "Grating type entered not in sequence with other gratings."
  GOTO GrtErrorPrinted

WaveNotInSeq:
  CLS : BEEP
  PRINT "Cut-on wavelength entered not in sequence with other gratings."

GrtErrorPrinted:
  PRINT : PRINT Dashes$: WaitMessage$ = "Press spacebar to continue."
  GOSUB SpaceToContinue
  GOTO NewGrt

ChangeGrtParams:
  GrtnGPrm#(POST, 1) = CHANGE1
  GrtnGPrm#(POST, 2) = CHANGE2: GrtnGPrm#(POST, 3) = CHANGE3
  GOTO NewGrt

StoreFlGrt:
  ' STORE CHANGES
  ON ERROR GOTO UnknownGrtStoreError
  CLOSE #2: OPEN "GR-FL.DAT" FOR OUTPUT AS #2
  ON ERROR GOTO 0
  WRITE #2, NumFltrs%
  FOR I = 1 TO NumFltrs%
  WRITE #2, FltrWvlnGth#(I)
  NEXT I
  WRITE #2, NumGrtnG%
  FOR I = 1 TO NumGrtnG%
  WRITE #2, GrtnGPrm#(I, 1), GrtnGPrm#(I, 2), GrtnGPrm#(I, 3)
  NEXT I
  CLOSE #2
  GOTO DisplayGratingDefaults

UnknownGrtStoreError:
  GOSUB AlertUserOfError 'Unidentified ERROR
  RESUME
```

DRAFT

```
' -----TIMELOOP MEASUREMENT SUBROUTINE-----
TimeLoopMeasurement:
PRINT
PRINT "Calibrating measurement delay loops against system time, please wait ...."
PRINT : PRINT Dashes$: PRINT
IPEAK = 1000 ' sets the upper count limit for the slowest machine
DEF SEG = &H40

FindPeriodBeginning:
LB1 = PEEK(&H6C): LB2 = PEEK(&H6C)
IF LB2 = LB1 THEN FindPeriodBeginning
' above is to find the beginning of the period
LB1 = PEEK(&H6C): NB1 = PEEK(&H6D) ' start count
FOR I = 1 TO IPEAK: NEXT I ' timing loop
LB2 = PEEK(&H6C): NB2 = PEEK(&H6D) ' finish count
C1 = NB1 * 256 + LB1: C2 = NB2 * 256 + LB2 ' computes times
IF C2 < C1 THEN GOTO FindPeriodBeginning
' in case of 256 turn-over
IF C2 - C1 >= 20 THEN GOTO CalcTimeLoop ' 20 counts gives an accuracy of 5%
IF C2 - C1 = 0 THEN IPEAK = IPEAK * 2: GOTO FindPeriodBeginning
' above -- if the count is 0 then count doubles and tries again
IPEAK = IPEAK * 21 / (C2 - C1) ' ratio to try again, should take 1 try
GOTO FindPeriodBeginning

CalcTimeLoop:
T# = CDBL((C2 - C1) / 18.2) ' 18.2 COUNTS/SEC
MSM# = CDBL(IPEAK / T# * .001)' loop/msec TIMING VARIABLE
CLS : RETURN
```